基于事件处理的分布式系统故障定位技术

杜翠兰[1] 谭建龙[2] 王晓岩[2,3] 张 宇[2,3] 刘 萍[2] 樊冬进[1]
(国家计算机网络应急技术处理协调中心 北京 100029)[1] (中国科学院信息工程研究所 北京 100093)[2]
(中国科学院大学 北京 100049)[3]

摘 要 近年来,分布式计算系统的规模越来越大、行为越来越复杂难控,系统中出现的各种故障也呈指数级增长,造成了非常严重的危害和损失,并且出现问题时对故障的排查、定位难度进一步加大。传统的通过跟踪程序运行轨迹来判断程序运行正确与否的方法,在分布式监控信息的交互上因消耗过大而且对目标程序侵入性高,已经难以满足软件行为分析的需求。通过复杂事件的处理及时发现和定位系统故障在事件大量、快速、不间断发生的分布式监控环境中显得尤为迫切。它可以利用有意义的信息状态变化事件分析系统行为,进而判断系统的运行状况,及时发现系统故障并定位,保证系统的健康运行。当前已有的复杂事件描述语言大多数是基于SQL的方法来描述复杂事件。这种数据流查询语言对于普通用户而言比较复杂,难以掌握。通过构建一种基于集合的事件流模型,对事件进行形式化定义,使用集合来表示事件,并定义相应的操作,使得用户只需掌握几个简单的集合操作,便可以定义复杂的故障规则。

关键词 分布式网络,实时监控系统,故障定位
中图法分类号 TP393 **文献标识码** A

Fault Location Technology Based on the Distributed Event Processing System

DU Cui-lan[1] TAN Jian-long[2] WANG Xiao-yan[2,3] ZHANG Yu[2,3] LIU Ping[2] FAN Dong-jin[1]
(National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing 100029, China)[1]
(Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China)[2]
(University of Chinese Academy of Sciences, Beijing 100049, China)[3]

Abstract In recent years, distributed computing systems become larger and more complex to control. System faults are growing exponentially, resulting in a very serious harm and loss, and problems on trouble shooting and positioning difficulty further enlarges. Traditional ways by tracking program to judge the running and correct method, using excessive consumption of the target program and invasive in distributed monitoring information interaction, has been difficult to meet the demand of software behavior analysis. Through the complex event processing in time to find and locate the fault, this need in events in a large, rapid, uninterrupted occurrence of distributed monitoring environment appears especially urgent. It can use the meaningful information state change events to analyze system behaviors, and then judge the system operating conditions, to detect fault and positioning system, ensure the healthy operation. The complex event description language is based on the SQL method to describe the complex events. This data stream query language is complex for ordinary users and difficult to master. By constructing a set based event flow model, we can use the set of events to conduct a formal definition. The user only needs to master a few simple assembly operations in order to define complex fault rule.

Keywords Distributed network, Real-time Monitoring system, Fault location

1 研究现状

1.1 故障定位技术

故障是软、硬件的缺陷,错误则是软、硬部件的不正确输出,失效是指所有和某故障有关的错误造成的网络的非正常运行。一个故障是若干错误的直接或间接的原因,错误是故障的表现,失效是故障的总效应。某部件的错误不一定由于内部存在故障,在网络环境中更有可能是由于故障的传播所导致的。故障、失效和告警事件之间的关系可以用图1表示。

故障定位一般包括故障检测、事件过滤、故障定位和故障统计分析4个步骤[1],各步骤的具体功能和流程如下:

本文受国家"242"信息安全计划基金项目(2010A029),中国科学院战略性科技先导专项(XDA06030200)资助。
杜翠兰(1966-),女,硕士,高级工程师,主要研究方向为网络信息安全,E-mail:dcl@isc.org.cn;谭建龙(1974-),男,博士,研究员,主要研究方向为算法设计、数据流管理技术研究、信息安全;王晓岩(1989-),女,硕士生,主要研究方向为网络信息安全;张 宇(1987-),男,博士生,主要研究方向为网络信息安全、模式匹配;刘 萍(1972-),女,硕士,助理研究员,主要研究方向为模式串匹配技术、算法设计;樊冬进(1983-),女,博士,工程师,主要研究方向为计算机应用、图像处理等。

故障检测：故障检测的目的即是在故障发生以后，尽可能将其识别出来。这一阶段的输入是代理报告关于网络资源改变的信息。

事件过滤：故障定位系统应提供过滤器和阈值机制以过滤过量的信息。通过设置过滤器过滤掉不重要和不关心的事件、重复告警噪音等，找出需要处理的事件。

故障定位：故障定位功能的目的是确定网络中故障的设备位置，甚至具体到发生故障的软件系统，这是故障管理的难点。

统计和分析：故障定位系统应支持故障记录、统计和分析，例如故障发生频率、哪些故障影响提供的服务等，还包括故障管理系统自身性能的分析统计，例如故障识别率等。

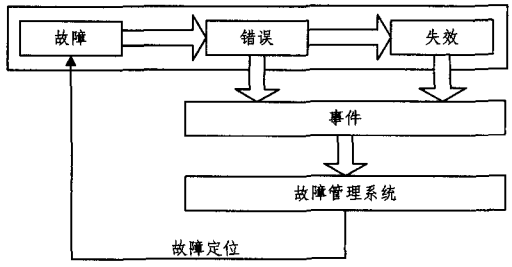

图 1　故障、失效和告警事件关系示意图

1.2　事件关联技术

事件关联(Event Correlation)技术是全新的故障管理策略，简单地说，事件关联就是对观测到的异常事件从语义上通过相关算法发现并定位真正故障原因的过程。事件关联过程寻找故障源，对于无法确诊的故障，事件关联步骤排除由网元相关性和依赖性引发的冗余事件信息，提供精简的症状视图给网管人员，以利于网管人员作进一步的故障诊断。

网络故障定位主要由事件的监视、解释和处理(响应)这3部分组成。网络中被管对象状态的改变，通常表示网络或软件运行过程中出现的异常情况。事件通常是底层问题的直接或间接反映，例如网络硬件或软件实效、性能瓶颈、配置不一致或非法的入侵尝试等。由于某个网络资源(主机、路由器、集线器、系统和应用程序等)的单一问题事件可能导致影响相关设备或子系统的正常操作，因此可能引起相关资源中许多症状事件的产生。当一些网络失效发生时，网络管理员常常会被大量事件消息"淹没"，这样的情况在网管领域中被称为"事件风暴"[2]。大量的事件消息使网管员无法过滤和正确地解释这些信息，继而导致问题诊断更长的延迟和带来更大的商业损失。因而要求故障管理系统能够帮助网络管理人员对观测事件进行关联以鉴别和定位底层问题。

目前事件关联技术的研究主要有以下几种方向。

1.2.1　基于规则的推理(Rule-based reasoning)技术

基于规则的推理技术是把告警相关性知识总结为一组相关性规则集，规则的形式为：IF condition THEN action。基于规则的方法的优点是直观，比较符合人们的思维习惯，便于理解。但是这种方法的问题在于：(1)当规则的数量达到一定的规模时，规则库的管理和维护将变得十分困难；(2)基于规则的相关性分析同样存在着知识获取的瓶颈；(3)规则不能适应网络拓扑结构以及网络配置的变化；(4)缺乏记忆性也是基于规则的相关性分析的一个主要限制。由于没有充分利用过去的经验，即使同样的情况再次出现，系统也要从成千上万的规则中去查找，严重影响了系统的工作效率。ECS[3]是惠普公司开发的一个基于规则的实时相关性处理系统，它包括输入、输出、过滤、延时、计算、组合、更改等。ECS通过建立网络模型实现告警事件之间的动态评价，通过构件来实现告警相关性分析和告警过滤，通过模块间的不同组合来实现不同的功能，以适应不同网络的实际情况。此外，基于规则的推理技术的商用系统也已经有所建立，包括 Computer Associates TNG 和 Tivoli TME。

1.2.2　基于案例的推理(Case-based reasoning)技术

基于案例的推理技术是通过利用过去的经验和方法解决新出现的问题。在基于事例的推理系统中，过去解决问题的经验都是以事例的形式存放在事例库中，当遇到新问题时就从事例库中寻找相同或相似的事例，通过对该事例的修正去解决新问题，同时解决新问题的经验又作为新的事例被添加到事例库中。事例库的维护主要按照遗忘曲线理论，即长期不用的信息将会被遗忘，所以需要删除长期不用的事例。Lewis 曾经设计实现了一个基于事例推理的故障追踪系统 CRITTER[4,5]。在 CRITTER 中，系统会对每一个发生的故障产生一个故障清单，故障的解决方案总是和故障清单一起保存到系统事例库中。当系统发生故障时，首先生成其故障清单，然后从事例库中寻找相似的故障清单，提出解决方法，并把新的故障清单和其解决方案添加到事例库中。

1.2.3　基于模型的推理(Model-based reasoning)技术

基于模型的推理技术通过建立网络模型来对网络的行为进行推理。网络模型主要包括网络结构信息(如网元类型、网络拓扑、包含的约束等)和网络行为信息(如告警相关性分析的动态过程)。作为模型组件的表现，传统的面向对象模型本身具有属性，同时与其他对象关联，而且具有行为。对象间的关系与模型间的关系相似。基于模型的相关性分析系统对新的故障具有一定的分析能力，但当处理超出其知识范畴的问题时，系统的性能将显著下降。IMPACT[6]是 GTE 实验室开发的一个典型的基于模型推理的系统，它用于固定和移动通信网络的告警相关性分析。IMPACT 在相关性规则的触发条件中考虑到了事件之间的时序关系，它同时引入了相关性窗口和活动周期。除了告警相关性规则外，IMPACT 提供了规则接口，领域专家可以自定义相关性规则。

1.2.4　基于代码本的推理(Codebook Based Approach)技术

基于代码本的推理技术[7]的基本思想是：对于每一种告警，可以看作是产生该告警的故障的特征，将其编码为该故障的特征向量。特征向量的每一维元素表示该故障对应的某类告警是否发生，用 0 或 1 表示，所有故障的特征向量一起构成了代码本，从本质上讲，代码本就是一个症状矩阵。建立代码本后，相关性分析的过程就是一个解码的过程。对于当前的故障，为其建立起特征向量，然后与代码本上的各个特征向量进行比较，计算当前特征向量与已知故障的特征向量之间的汉明距离，选择距离最小的特征向量对应的故障作为当前的故障。基于代码本的相关性分析方法通过对告警知识模型的预处理减少了实时告警相关性分析的复杂性，因此具有更高的效率。另外，对故障的确定是基于最小距离而不是严格的特征匹配，因而增强了系统的鲁棒性。但是，在实际网络中，"问题"和"征兆"往往比较复杂，数据量很大，进行有效编码以

获得最优的代码本非常困难，另外代码本技术也不具备自学习能力。

2 基于复杂事件处理的故障定位技术

分布式网络系统本身结构复杂，出现问题时对故障的排查、定位难度大[8]。传统的跟踪程序运行轨迹来判断程序运行正确与否的方法，分布式监控信息进行交互时消耗过大，而且对目标程序侵入性高，已经难以满足软件行为分析的需求。通过复杂事件的处理及时发现和定位系统故障在事件大量、快速、不间断发生的监控环境中显得尤为迫切。它可以利用有意义的信息状态变化事件分析系统行为，进而判断系统的运行状况，及时发现系统故障并定位，保证系统的健康运行。

当前已有的复杂事件描述语言大多数是基于 SQL 的方法来描述复合事件。数据流查询语言重点关注数据而非事件的组合；很少关注排序等其他时间关系，通常对表示发生时间的字段进行排序来识别单数据流的时序关系，采用连接、选择等 SQL 操作子识别多数据流的时序关系[9]。最为典型的数据流查询语言是 CQL，CQL 在数据流上应用 SQL：对每个时间点，利用滑动窗口这种流到关系转换的操作，将所有接收到的数据流都转换为关系；之后的查询评估都被作为普通的 SQL 查询。目前，数据流查询语言不提供流到流的转换操作，因此扩展数据流查询语言描述复合事件，通常都提供流到流的操作子，例如顺序、逻辑与等。这种数据流查询语言对于普通用户而言比较复杂，难以掌握，于是我们提出了一种基于集合的事件流模型，其将事件进行了形式化定义，用集合来表示事件，并定义了相应的操作。这样用户只需掌握几个简单的集合操作，便可以定义复杂的故障规则。下面给出事件和操作的详细定义。

为了方便说明，定义了如表 1 所列的若干符号，下面对这些符号进行详细的说明。在系统中，我们会采集系统运行时产生的若干信息，比如：网络流量、丢包率、CPU 使用率等。每一种信息表示一种属性，用 a 表示，对于每一个 a 都有一个对应的属性值，用 v 表示。用一个属性名称和其对应的值表示一个属性值对，记为 p。事件用 p 的集合来表示，事件集合记为 R。故障定义为满足特定条件的事件集合，记为 F。比如：每一分钟记录一次 CPU 的使用率，那么每分钟就会产生一个事件，每个事件包括多个属性值对。表 2 为我们采集的一条 CPU 使用率事件的结构。

表 1 符号定义表

符号	定义
t	事件
p	事件属性值对
a	事件属性名称
v	属性值
R	事件集合
V	故障

表 2 CPU 使用率事件结构

属性(a)	值(v)	含义
ID	1	唯一区分事件标示
Rate	95%	CPU 使用率
IP	192.168.0.100	主机 IP 地址
Time	2012-12-01 22:01	时间采集时间

对事件集合的操作有 4 中，分别为：选择操作、连接操作、分组操作和投影操作，下面介绍这 4 种操作的定义。

1)选择操作

$$\sigma C(R)=\{t \mid t\in R \wedge C(t)=\text{true}\} \tag{1}$$

式中，σ 表示选择操作，R 是选择操作作用的事件集合，C 是选择条件，可以描述成对某几种属性对应的属性值的约束。该运算的结果是一个事件集合。例如：CPU 使用率事件集合中超出使用率大于 95%的事件集合。

2)连接操作

$$t_1 J t_2=\{pt_1, pt_2 \mid pt_1\in t_1 \wedge pt_2\in t_2 \wedge J(pt_1, pt_2)=\text{true}\} \tag{2}$$

式中，J 为连接条件，pt_1 和 pt_2 分别是事件 t_1 和 t_2 的某些属性值对，通过对这些属性值对对应的属性值进行某种运算，得出符合条件的新的属性值对的集合，也就是新的事件。例如：根据 CPU 使用率事件和某个特定进程内存占用率事件，选择相同时间和相同 IP 地址的事件中，当 CPU 占用率大于某个阈值时，都有哪些进程在运行。

3)事件集合分组操作

$$GroupG(R)H=\{t_1, t_2 \cdots \mid G(p)=\text{true and } H(t)=\text{true}\} \tag{3}$$

式中，G 是事件集合分组条件。将事件集合 R 按照 G 的值划分为多个子事件集合，并且子集合满足条件 H(H 可以为空)，运算结果是一个新的事件集合。例如：找出所有 CPU 使用率事件中使用率大于所有 CPU 使用率事件的平均值的那些事件集合。

4)事件投影操作

$$\Pi P(t)=\{p \mid p\in t \text{ and } P(p)=\text{true}\} \tag{4}$$

式中，p 为事件 t 的事件属性值对，P 为投影条件。这个操作就是将一个事件的属性值对的个数减少，产生新的事件，新的事件的属性值对是原事件属性值对的子集。在特定的一个运算里，得出的最终的故障事件可能不必包含所有的属性值对，可以利用这个操作除去冗余的信息。

3 系统结构

根据我们提出的基于复杂事件处理的故障定位技术，实现了一个分布式复杂事件处理(Distributed Complex Event Processing)原型系统 DSCP。该系统自下而上由 3 个层次构成：事件信息采集层、事件监控分析层和系统显示层，如图 2 所示，各层功能分别为如下。

事件信息采集层：该层通过分布式的事件采集代理采集分布式网络系统的事件信息，并写入事件流数据库。

事件监控分析层：该层包括检测规则自动转换和事件流监控\分析两个子模块。我们定义了一种基于集合的事件流过滤模型，检测规则自动转换子模块根据该模型定义对规则数据库中的记录进行自动转化，生成 SQL 语句。事件流监控\分析子模块读取自动转化生成的 SQL 语句，扫描事件流数据库中的记录，监控、分析出分布式系统的故障。

系统显示层：该层包括故障报警、故障查询、检测规则配置、事件信息实时查看 4 个子模块。故障报警子模块实时展示时间监控分析层定位出的分布式网络系统故障。故障查询子模块接收用户输入的分类查询信息，查询故障数据库后显示查询结果。检测规则配置子模块对不同用户开放不同接口，将用户的输入写入规则数据库。事件信息实时查看子模

块将时间采集层采集到的事件流实时展示在系统界面上。

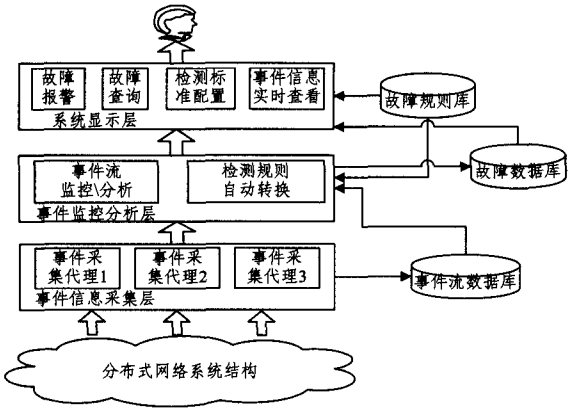

图 2　故障定位原型系统架构示意图

数据采集服务器从分布式网络系统中采集各种事件流信息，并写入事件流数据库。事件监控分析服务器根据规则数据库中配置的检测规则，自动生成用于事件监控检测的 SQL 语句，对事件流数据库中的数据进行检测，并将检测出的故障及相关信息写入故障数据库。检测规则配置、故障报警和显示均在负责系统显示的主机上完成。

4　应用实例

我们选取了一个应用中的分布式网络系统，这个系统中经常出现的故障包括：程序输入包与网卡输入不一致、链路拥塞、DNS 攻击检测、分流负载不均等。通过对这个分布式网络系统故障的综合分析，我们选取 3 种典型的系统故障，对开发的故障定位系统 DCEP 进行功能测试，通过在这个分布式网络系统的每个节点上采集系统的日志信息，验证提出的基于复杂事件处理的故障定位技术的有效性，具体的部署情况如图 3 所示。

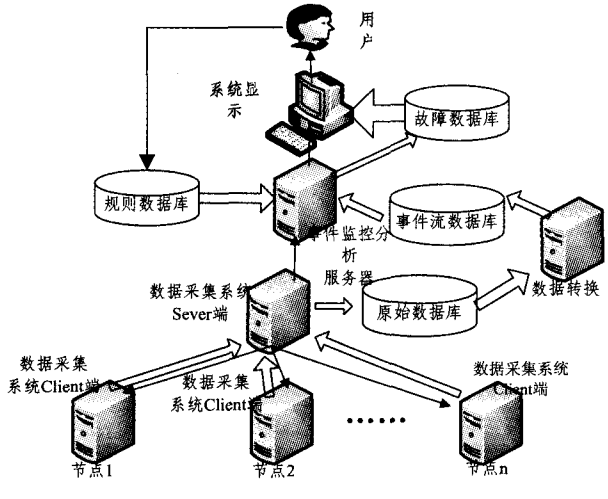

图 3　故障定位系统 DECP 在分布式网络系统中的部署示意图

我们采用 C/S 架构实现采集模块，将 Client 端部署到分布式网络系统的各个节点上，Sever 端部署到单独的数据采集系统服务器，Sever 端下发命令到 Client 端，由 Client 端采集业务系统日志文件信息，并返回数据给 Sever，Sever 将数据存入原始数据库，数据转换服务器将原始数据转换为事件流数据。检测模块包括事件监控分析服务器，通过事件流数据库和规则数据库得出检测结果并存入故障数据库。故障显示客户端通过查询故障数据库将故障显示给用户。

这里需要注意的是，分布式网络系统需要开放本地的日志，并且采用统一的格式，以便数据采集系统 Client 端采集数据。

我们选取的 3 种典型故障为：程序输入包与网卡输入不一致、链路拥塞定位和 DNS 攻击检测。以下分别阐述故障定位系统在检测 3 种故障的具体方法。

4.1　程序输入包与网卡输入不一致

分布式系统中经常发生丢包问题，当发生这一问题时会造成程序输入包与网卡输入不一致的故障，为了检测这种故障，我们在分布式系统中的每个节点上采集网卡输入的流量事件以及程序的输入流量事件。用基于集合的事件流过滤模型检测这种故障，如表 3 所列。

表 3　程序输入包与网卡输入不一致检测

操作	条件
选择操作	无
连接操作	网卡输入流量 ＞ 程序输入流量 * 1.2 OR 程序输入流量 ＞ 网卡输入流量 * 1.2
分组操作	无
投影操作	事件 ID 网卡输入流量 主机 IP 地址 程序名称 事件时间

在检测程序输入包与网卡输入不一致这种故障时，我们通过统计以往的故障历史数据，得出当网卡输入流量大于程序输入流量的 1.2 倍时，或者程序输入流量大于网卡输入流量的 1.2 倍时，就会产生这种故障。不同的系统肯定有不同的判断标准，可以通过统计学习的方法修改阈值，以提高故障检测的准确率。

4.2　链路拥塞定位

链路拥塞是指这个分布式网络系统中某些链路流量过大的现象。在正常工作的系统中，具有相同功能的不同节点处理的数据量应该是比较接近的，这称为负载均衡，当出现链路拥塞故障时，就会导致某些节点的处理量过大。用基于集合的事件流过滤模型检测这种故障，如表 4 所列。

表 4　程序输入包与网卡输入不一致检测

操作	条件
选择操作	无
连接操作	无
分组操作	网卡输入流量 ＞ 所有节点网卡输入流量平均值 * 2
投影操作	事件 ID 网卡输入流量 主机 IP 地址 事件时间

在检测链路拥塞定位时，网络的总流量是变化的，我们无法提前给出阈值，可以通过分组操作，找出网卡输入流量大于平均流量 2 倍的那些节点。其中，我们选取的是大于平均流量 2 倍的那些节点，这个倍数也可以通过统计学习的方法来进行修改，以提高准确率。

4.3　DNS 攻击检测

分布式网络系统可能遭受外来的 DNS 攻击，发生 DNS 攻击时，这个分布式网络系统中的 DNS 包数量会发生异常，我们在系统中的节点上采集。通过检测 DNS 包数量事件可以检测出 DNS 攻击。类似前两种的检测方法，用基于集合的

事件流过滤模型表示，如表5所列。

表5 程序输入包与网卡输入不一致检测

操作	条件
选择操作	DNS包数量 > 阈值
连接操作	无
分组操作	
投影操作	事件ID 网卡输入流量 主机IP地址 事件时间

5 性能测试

5.1 测试数据集说明

根据程序输入包与网卡输入不一致这种故障发生时分布式网络系统相关数据指标的特点，人工构造50组故障数据，将这些随机插入采集的真实网络流量中，构成最终用于系统性能测试的模拟数据集。真实网络流量的采集方法和模拟故障数据的构造方法如下：

真实网络流量：通过DCEP的分布式采集节点，我们连续24个小时采集这个分布式网络系统的事件流信息。

模拟故障数据：这个分布式网络系统在正常运行时，网卡出入流量和程序输入流量的数值差不应超过20%。我们统计分析真实数据集，得出一个月来程序输入流量的最大值和最小值，分别记为max和min。在构造每组故障数据时，通过rand()函数生成区间[min,max]上的伪随机数，将该值作为程序输入流量的数值。然后，通过rand()函数生成区间(0.70,0.80)或者区间(1.20,1.30)上的伪随机数，将该值记为rate。程序输入流量 * rate的数值作为网卡输入流量的数值。

5.2 系统性能测试结果与分析

通过我们编写的模拟器程序重放测试数据集数据，模拟分布式事件流信息采集代理定时采集某个分布式网络系统事件流信息的过程，性能测试结果与分析如下：

1)系统在运行时检测出测试数据集中的"输入包与网卡输入不一致"故障49个，并将发生故障的设备和运行的程序准确定位，正确率达到98%。

2)系统检测出"输入包与网卡输入不一致"故障并进行报警共计用时44s，具有较高的时效性。

结束语 面对分布式网络系统对系统故障及时发现并准确定位的应用需求，我们提出基于复杂事件处理的故障定位方法。在此基础上，我们开发了故障定位系统DCEP，用以验证基于复杂事件处理的故障定位方法的可行性。我们对故障定位系统DCEP的功能进行了测试，测试结果显示，该系统具有较高的故障检测准确率和较快的时效性，进一步证明了基于复杂事件处理的故障定位方法的有效性。DCEP仅为理论验证性的原型系统，在日后的研究中可以对系统功能进行进一步的研究和完善。

参考文献

[1] Kamoshida Y, Taura K. Scalable Data Gathering for Real-Time Monitoring Systems on Distributed Computing[C]// Proceedings of IEEE International Symposium on Cluster Computing and the Grid. Tokyo, Japan, IEEE Computer Society, May 2008

[2] Robert D, Gardner David A. Network Fault Detection: A Simplified Approach to Alarm Correlation[C]// Proceedings of XVI World Telecom Congress, university of Strathclyde. 1997: 115-123

[3] Harrison K. Event Correlation in Telecommunication Network Management[R]. Hewlett-Packard Labs, Bristol, 1994

[4] Lewis L. A Case-based Reasoning Approach to the Management of Faults in Communication Networks[C]// Proceeding IEEE Infocom'93, vol. 3. San Francisco, 1993: 114-120

[5] Lewis L. Implementing Policy in Enterprise Network[J]. IEEE Communications Magazine, 1996, 34(1): 50-55

[6] Jakobson G, Weissman M. Alarm Correlation [J]. IEEE Network, 1993, 7(6): 52-59

[7] Gabriele S, Chiaravalloti E, D'Aquila Q, et al. Distributed real-time monitoring system to natural hazard evaluation and management: the AMAMiR system [C]// Proceedings of World IMACS|MODSIM Congress. 2009

[8] White W, Riedewald M, Gehrke J. What is "next" in event processing[C]// Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems. New York, NY, USA, 2007: 263-272

[9] 岳海涛. 基于事件关联和数据挖掘的网络故障管理技术的研究[D]. 长沙：中南大学，2010

(上接第270页)

[8] Bondy J A, Murty U S R. Graph Theory with Applications[M]. London and Basingstoke: MacMillan Press LTD, 1976

[9] Harary F. Recent results and unsolved problems on hypercube theory[M]. Alavi Y, Chartrand G, Oellermann O R, et al., eds. Graph Theory, Combinatorics, Applications, John Wiley & sons, 1991: 621-632

[10] Barnes G H, Brown R M, Kato M, et al. The ILLIAN computer [J]. IEEE Transactions on Computers, 1968, 17: 746-757

[11] Kai Hwang. 高等计算机系统结构：并行性、可扩展性、可编程性[M]. 王鼎兴，等译. 北京：清华大学出版社，南宁：广西科学技术出版社，1995

[12] 李亚民. 计算机组成与系统结构[M]. 北京：清华大学出版社，2000

[13] Witte D, Gallian J A. A survey Hamilton cycles in Cayley graphs [J]. Discrete Mathematics 1984, 51: 293-304

[14] Curran S J, Gallian J A. Perspectives Hamiltonian cycles and paths in Cayley graphs and digraphs-A survey [J]. Discrete Mathematic, 1996, 156: 1-18

[15] Alspach B, Bermond J C, Sotteau D. Decompositions into Cycle: Hamilton decompositions [M]. Hahn G, ed. Cycles and Rays (kluwer Academic pubbishers, Netherlands), 1990: 9-18

[16] Araki T, Kikuchi Y. Hamiltonian laceability of bubble sort graphs with edge faults [J]. Information Sciences, 2007, 177: 2679-2691

[17] 师海忠，路建波. 关于互连网络的几个猜想[J]. 计算机工程与应用，2008, 44(31): 112-115

[18] 高随祥. 图论与网络流理论[M]. 北京：高等教育出版社，2009